

REMARKS

In Response to the Examiner's Office Action of March 9, 2004, Applicant is now providing the following comments and amendments to this application.

Claims 1, 4, 8, 12, 14, 16 and 17 are presently pending in this application.

In regard to the Examiner's objection to Applicant's Declaration, Applicant is now providing a new Declaration in compliance with 37 CFR 1.67(a). This Declaration now properly states --- that the person making the oath or declaration acknowledges the duty to disclose to the office all information known to the person to be material to patentability as defined in 37 CFR 1.56.

In regard to the drawings, Applicant has now corrected Figs. 1-5 so that they designate the appellation of "prior art". Additionally, Figs. 7C and 7D have been corrected to indicate "a first two-dimensional buffer array" instead of --one-dimensional array--.

In regard to the Examiner's objections to the drawings, Applicant has corrected the errors in Fig. 6 to indicate that the last column of the array should be labeled --8191--. This is likewise true for the ARRAY BUFFER. Further, the upper left-hand corner reference in Fig. 7C has now been corrected to read --I--.

In regard to the specification, the statement "1D" on page 7 line 4, has now been corrected to read --1B--. Further, the title has now been removed from the Abstract.

Likewise, page 24 line 33 has been corrected to read --8191-- instead of "8192".

Substantive Matters:

While Examiner has withdrawn his rejection of claims 1 and 4 under 35 USC 102(a)(e), and also Examiner's rejection of claims 8, 12, and 14, under 35 USC Article 103(a), has been withdrawn, it appears that Examiner has now instituted a new ground of rejection, where he has indicated that claims 1, 4, 12 and 14, are rejected under 35 USC 112, first paragraph, as failing to comply with the "enablement requirement". Examiner argues that the specification did not sufficiently describe the subject matter as to enable one skilled in the art to make or use the invention.

The Examiner contends that the subject matter of claim 1, clause (b), and also claim 4, clause (b2), is not sufficiently supported in the specification to enable one skilled in the art to make or use the invention.

At this juncture, Applicant would traverse such a conclusion by the Examiner. There is no warrant for making such a conclusion, since it is well-known for many years, in the software arts, that even rather unskilled software people know that at a certain point in the coded sequence of operations, there is a decision block or selection operation which is very commonly used to select one alternative path over another.

For example, in the Microsoft Press Computer Dictionary, Third Edition, Copyright 1997, there is indicated the concept of "branch instruction":

Branch Instruction: As assembly or machine level instruction that transfers control to another instruction, usually based on some condition (that is, it transfers if a specific condition is true or false). Branch

instructions are most often relative transfers, jumping forwards or backwards by a certain number of bytes of code.

Branch Point: A location at which a given branch instruction occurs if the attendant condition (if any) is true.

Thus, Applicant would indicate that is completely unwarranted for the Examiner to make such a conclusion regarding the ability of a skilled engineer to perform such operations --- since such selective operational factors are already well-known and commonly used in the software arts.

It is commonly known that, in the software sequence of operations, when a decision point or branching operation occurs, then at that point, a step is commonly applied to select one alternative operation over another, depending on what sort of input occurs at that particular branch point.

Therefore, Applicant contends that this type of operation is well-known in the art, and easily available to a skilled engineer to utilize such a practice in developing software for such a purpose. Thus, Applicant respectfully requests that Examiner withdraw his objection to the concept of "lack of enablement" under 35 USC, Article 112, first paragraph.

Also attached in Appendix I is a cover sheet and certain designated coding of a 1994 DFAST software patch utilized by Applicant's employer, Unisys Corporation, which patch was in common use by software engineers to utilize branch points which can select one path of operations or another path of operations at a given branch point.

Additionally, the attached Appendix I will indicate portions of the DFAST program in 1994 which will illustrate

engineering usage of the selection of first, and first and second buffer arrays.

Thus, regarding Applicant's claim 1, the Examiner cites Applicant's Figs. 7B and 7C contending that the disclosure does not support selection between a "single or dual" two-dimensional array means, as recited in limitation (b).

Here, it should be indicated that the decision of whether to use a single or a dual two-dimensional array means depends on an input to the branch point which indicates whether or not the single two-dimensional array will be sufficient or whether a particular number of bytes would require a dual two-dimensional array.

It is well-known in the computer art that a Central Processing Unit can assess the number of bytes to be downloaded and count them in order to allow the branch point to decide whether a single two-dimensional array will do, or if a dual two-dimensional array is needed.

In regard to claim 4, the Examiner contends that the disclosure does not appear to support selecting a buffer array size which most closely approximates (and will accommodate) the recognized number of bytes to be downloaded. Again here, it should be understood that the Central Processing Unit is commonly known to be able to count the number of bytes to be downloaded and from this, to select the appropriate number of buffer arrays needed to accommodate the number of bytes involved. These type of operations are already well-known in the state of the art.

Regarding Applicant's claim 12, Examiner says Applicant does not appear to support selection between a single or double two-dimensional array means, as recited in limitation (e).

Further, per claim 14, Examiner states the disclosure provides support for using two 384 KB two-dimensional arrays, but it does not appear to support other "selected sizes" of these arrays. Here, Applicant would state that other "selected sizes" of arrays (i.e., one or more two-dimensional arrays) can easily be calculated by the Central Processing Unit in order to make a decision as to how many two-dimensional buffer arrays would be appropriate for selection.

Regarding claim 4, in line 2 of limitation (b), the subject of "said firmware" has now been changed to read --- said SCSI firmware and SCSI servo firmware ----.

Regarding claim 8, Applicant has now deleted the parenthetical phrase (USERMAINTREQUEST).

Regarding Applicant's claim 8, lines 4-5, in limitation (b), and again in lines 1-2 of limitation (c), where the term is used "said SCSI firmware and said SCSI servo firmware" -- this has now been corrected so that the term "microcode firmware" in limitation (a) has now been changed to read --- SCSI firmware and SCSI servo firmware ----.

In regard to Applicant's claim 14, the statement of "said SCSI firmware data" in line 1 of clause (d) and line 1 of clause (e), these statements have now been changed to read --said SCSI firmware--.

In Applicant's claim 16, with regard to Examiner's comments regarding the open-ended limitations as being unclear, these limitations have now been restated in each case in order to clarify them so that there is no doubt as to their meaning.

In regard to Applicant's claim 1, clause (b), it should be noted that in Fig. 7B at step (vn), a question arises as to whether the firmware is greater than 393,216 bytes --- then, if the answer is YES, it will be seen that at step (vo), the "first" two-dimensional buffer array is used by means of issuing Write buffer commands with a block of 8192 bytes of data. This goes through continuation marker I over at Fig. 7C, whereupon then a second two-dimensional buffer array is used at step (vq1).

Thus, it is seen when there are more than 393,216 bytes, it will be necessary to use "two" separate two-dimensional buffer arrays, as seen in steps (vo) and (vq1).

Now, returning to Fig. 7B at step (vn), if the answer is NO, that is to say, the firmware involves less than 393,216 bytes, then the sequence goes through continuation marker H over to Fig. 7C, where at step (vm1) there is only required the use of a first or "single" two-dimensional array.

Fig. 7C has now been corrected at step (vm1) to read as follows: "USE SINGLE TWO-DIMENSIONAL ARRAY" --- as the previous statement of "USE ONE-DIMENSIONAL ARRAY" was misleading and was intended to indicate that a single two-dimensional array was used. The two-dimensional array is shown in Fig. 6, and the present invention allows the choice of utilizing one of these two-dimensional arrays, or using two of these two-dimensional arrays depending upon the number of bytes of firmware to be downloaded.

Likewise now, Fig. 7D at step (ivg3), which previously stated "USE ONE-DIMENSIONAL BUFFER ARRAY" --- this has now been changed to read "USE SINGLE TWO-DIMENSIONAL BUFFER ARRAY".

This clarification was in the original specification on page 6, "SUMMARY OF THE INVENTION", at line 5, where it was indicated --- using a set of two specialized two-dimensional arrays to overcome the limitations of systems array capacity. Likewise, at page 6, lines 19-20, the specification indicates --- through the use of a dual two-dimensional system capacity arrays ---.

Then, at page 6 line 25, it was seen in the original specification --- a selection is made as to use of a single two-dimensional array or a dual two-dimensional array for buffer loading.

Thus, dependent upon the number of bytes to be downloaded (as seen in Fig. 7C, step (vn)), if the firmware is greater than 393,216 bytes, the sequence will proceed to use a first two-dimensional buffer array and then use a second two-dimensional buffer array.

Then, at step (vn), if the firmware is less than 393,216 bytes, the sequence proceeds to Fig. 7C, where there is used a single (or first) two-dimensional array.

It is regretted that the earlier statement in the sequential blocks (which indicate a one-dimensional array) was inappropriately labeled, and should have said "single two-dimensional array".

Because of this newly-allowed greater buffer array capacity, it was now possible to download an exceedingly greater number of data bytes for downloading, while in previous configurations, no such availability of two dual two-dimensional arrays were available.

It should be noted that these buffer arrays which are set up, are done through software as "virtual arrays" and do not involve hardware, RAMs or ROMs. Thus, considerable saving of hardware configurations are involved in this situation.

The Central Processing Unit in the subject invention is typically that of the Unisys Corporation A-Series computer systems which are used to download the firmware. The Unisys Master Control Program (MCP) can only send a Read or a Write command with a maximum data length of 393,216 bytes.

Most of the old disk drive firmware used in Unisys Corporation's OEM disks was much less than the 393,216 bytes that could fit a single array and which could be constructed via software, and which could not be sent in one Read or Write command.

Newer firmware was developed that was larger than what the Unisys CPU could support, so that a new way of sending and downloading the firmware had to be designed. In order to fulfill this new requirement, there were developed multiple two-dimensional arrays which were constructed in the maintenance software of the CPU. A Read or a Write command had to have a continuous open state in order to download the firmware in 8192 byte chunks to the disk drive. As a result, there was developed a Central Processing Unit with software programmable selection means for choosing a single or a dual two-dimensional array means for temporarily storing said SCSI firmware, during the downloading process.

The "enablement" for making and using selectively a single two-dimensional array, or a dual two-dimensional array, will be seen in Appendix I in the attached portions of the DFAST program which illustrates how selectivity can occur through software means in choosing a single two-dimensional array or a dual two-dimensional array.

It is regretted that the original mis-wording in Figs. 7C and 7D were somewhat misleading and should have indicated that the utilization was for a "single two-dimensional array" (and not a one-dimensional array).

Thus, it should be clarified that the system uses a first two-dimensional array when there are less than 393,216 bytes --- and then on the other hand, the system will use a first two-dimensional buffer array and a second two-dimensional buffer array when there is greater than 393,216 firmware bytes to be downloaded.

In regard to Applicant's claim 1 clause (b), it should be noted that the language involves choosing a single or dual two-dimensional array means.

Note that this indicates a single two-dimensional array means, or a dual two-dimensional array means, and does not involve any one-dimensional arrays.

In regard to Applicant's claim 4 clause (b2), it should be noted that --- the means for selecting a buffer array size which most closely approximates said recognized number of bytes to be downloaded ---- this involves whether or not a single two-dimensional array is necessary or if there are a larger number of bytes, then there is necessity for selecting a first and a second dual dimensional array. Thus, the CPU software, by determining

the size of bytes to be downloaded and recognizing the size availability of each one of the buffer arrays, formulates a decision whether to use just one of the dual two-dimensional arrays, or whether to use a second two-dimensional array.

Thus, if the number of bytes to be downloaded is less than 393,216 bytes, then it is only necessary to use the first two-dimensional buffer array. On the other hand, if the number of bytes to be downloaded exceeds 393,216 bytes, then the system works to use not only the first two-dimensional array, but additionally uses the second two-dimensional array.

In regard to Applicant's claim 8 clause (f) which involves software selection means for selecting the appropriate size of said first and second two-dimensional buffer array means ---- this should be understood as just previously discussed, that depending upon the number of bytes to be downloaded, the method will use a single two-dimensional array, or the method will use the first and second two-dimensional arrays if there is a larger number of bytes to be downloaded.

In regard to Applicant's claim 12 clause (e) --- selecting a single or double two-dimensional buffer array appropriate to the byte count of said appropriately selected firmware for temporary storage --- here again, it should be understood that there is the availability of a single two-dimensional array, or a double, that is to say, a *second* two-dimensional array. Likewise, depending upon the number of bytes to be downloaded, it may be possible that the single two-dimensional array will fulfill the bill. But, on the other hand,

if there is a larger number of bytes to be downloaded, it may be necessary to use not only the first two-dimensional array, but also use a second two-dimensional array. This will be seen to be indicated in Figs. 7B at step (vo), which continues to Fig. 7C at step (vq1).

Likewise, Fig. 7C at step (vm1) has been corrected to more properly indicate the use of a single two-dimensional array, since the previous statement of a one-dimensional array was not properly articulated.

In the original specification, there was an inappropriate statement with the use of the term "one-dimensional array", since this should have said and indicated that this was the first two-dimensional buffer array of a group of dual two-dimensional buffer arrays. Thus, at certain times, the single two-dimensional array is sufficient for the download, while at other times it is necessary to use the first two-dimensional array and the second two-dimensional array in order to accommodate the length of the download bytes.

In regard to Applicant's claim 14 clause (d), it will now be seen that the downloading of the firmware involves using selected units (not sizes) of first and second two-dimensional buffer arrays. There are not any "selected sizes", but rather there is a selection of whether to use a first two-dimensional buffer array or a first and second two-dimensional buffer array.

In regard to Applicant's claim 16 clause (1), the clause now reads -- utilizing the download to a first two-dimensional buffer array --.

Now in regard to Applicant's claim 17 clause (dn6), this now reads --- setting up an adequate number of two-dimensional buffer arrays for a download ---.

Notice that in claim 17, clause (dn10) shows the utilization of the first two-dimensional buffer array, and then later the clause (dn12) shows utilizing the second two-dimensional buffer array.

It is regretted that some misleading language was originally shown in Figs. 7C and 7D, which listed a "one-dimensional array" which was inappropriately stated, when it should have indicated that there is a *first* two-dimensional array operating at certain times, and at other times, there is utilized a *first* two-dimensional array and a *second* two-dimensional buffer array. These clarifications have now been made to the specification and to the drawings.

Now, in regard to the amendments made to the various claims to coordinate them with Examiner's objections in phraseology, and further, in regard to the situation that indicates that the enablement factor in "branching operations" are well-known in the art and well-recognized in the art so that no undue experimentation would be required by a skilled engineer in order to practice Applicant's invention --- thus, Applicant would now request that Examiner survey the extant claims in order

to realize the functional value thereof, and as a result provide a suitable Notice of Allowance therefor.

Respectfully submitted,

By Alfred W. Kozak
Alfred W. Kozak
Reg. No. 24,265
(858) 451-4615
(949) 380-5822

Certificate of Mailing (37 CFR 1.8a)

I hereby certify that this paper (along with any paper referred to as being attached or enclosed) is being deposited with the United States Postal Service on the date shown below with sufficient postage as first class mail in an envelope addressed to: MAIL STOP AMENDMENT, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

Date:

May 26, 2004

Patti S. Freddy

Patti S. Freddy

APPENDIX I

Attached are selected pages of Coding indicating known engineering capability for showing:

- (i) software programmable selection means for choosing a single or dual two-dimensional array means —**
- (ii) means for selecting the appropriate number of array means of said first and second two-dimensional buffer array means —**

The implementation factors are shown in the attached pages D1, D2, D28, D29 and D30 in the attached excerpts of the DFAST program download in multiple chunks which was shown in documents with a 1994 copyright notice.

```

$$ VERSION 90.032.001                                00000092
$$ SET STACK LIMIT 100                                00010000
$$ RESET LIST XREF                                     00011000
$$ SET VERSION 01.005.000                             00012000
$$ SET OMIT                                             00013000
* PATCHFOR DOWNLOAD IN MULTIPLE CHUNKS                00013500010040006
*****00014000
*                                                     00015000
*   Class | Unisys                                00016000
*                                                     00017000
*   This material is restricted and proprietary to the 00018000
*   ***** Unisys Corporation and is not to be reproduced, shown, 00019000
*   ## or disclosed outside the Unisys Corporation. Customer 00020000
*   ## Services Engineering restricted and proprietary data 00021000
*   ## is furnished solely for use by Unisys personnel in 00022000
*   ## servicing customer's equipment. 00023000
*   ## 00024000
*   ## 00025000
*   ## This document is the property of and shall be returned 00026000
*   ## to Unisys Corporation, One Unisys Place, Detroit, MI 00027000
*   ***** 48232. 00028000
*                                                     00029000
*                                                     00030000
*   Material | Copyright (C) 1994. 00031000
*                                                     00032000
*                                                     00033000
*                                                     00034000
*                                                     00035000
*****00036000
$$ PAGE 00037000
*****00038000
*                                                     00039000
*   T A R G E T / F W L O A D 00040000
*                                                     00041000
*   Patch History 00042000
*                                                     00043000
*****00044000
4/94 Initial version. Will download A-code files to SBC controllers 00045000
and SCSI Disk Drives. 00046000
4/94 Changes - Qualification phase for official CSPO release 01.001. 00047000010010001
5/94 Add 'Express Mode' for engineering (DISKs only) release 01.002. 00047200010020003
6/94 Change location of some display statements. --- release 01.002. 00047500010020002
10/94 Fixed seg array error in Verifyfile procedure. release 01.002. 00047600010020004
00048000
*****00049000
      DOWNLOAD FIRMWARE to A-SERIES TARGETS (DFAST)
      =====
Utility's Part Number: 3492 4639 00055000
00056000
This utility's primary function is to load microcode to SCSI BUS 00057000
CONTROLLERS (SBC) and/or SCSI disk drives, and must be marked as a 00058000
PPed (Privileged Program) in order to operate. 00059000
00060000
Interaction with the user is via prompts at either a terminal or 00061000
an ODT. Terminal prompts use regular I/O. ODT inputs use the 00062000
ACCEPT statement. Otherwise the input rules are the same. The 00063000
program requires interactive user participation to execute properly. 00064000
00065000
00066000
Load: 00067000
----- 00068000
00069000
00070000
To operate, two basic elements must be present: an SBC controller or 00071000
SCSI disk drive, and a microcode file on disk or tape. The microcode 00072000
must reside on a unit served by a different controller or disk than the 00073000
one being initialized. If a "critical unit" exists on the string served 00074000
by the controller being initialized and there is only one path to that 00075000
critical unit, the system will reject the attempt to download the 00076000
microcode. 00077000

```

1) Step one is to determine the code file's capability by having the user enter a file name. Normal Family Substitution rules are in effect. If the file cannot be found, the user is prompted to enter another file name. The file name may include an "ON <family>" in the file declaration if it resides on disk (e.g., (UCODE)XYZ/123 ON MYPACK). If the file resides on tape, the file name only would be entered (e.g., SCZFRM).

If the code file is not a valid SBC or disk drive Acode file, the file is rejected and the user is prompted to enter another file name.

2) Step two is to determine the SBC controller or drive capability. The user enters a SBC or drive number and the utility reads the unit's attributes. If the SBC or drive number does not represent the correct target, or if the target attributes do not match those of the Acode file, the user is prompted to enter a new target number or 'Quit'. The SBC controller or drive must be reserved.

>>>> There may be no way to prove that the SBC controller controls any given drive or that it is the only path to the drive or that another path exists! The User should do an "OL" on the ODT to verify the paths available for the target.

Verifyfile:

Verify is simpler than Load since no SBC or drive is involved. The user answers the file name prompt and the Verify routine used by Load is called to generate a report on the attributes associated with that file.

Verify allows the user to cycle through multiple, potential A-code files until 'Quit' is entered.

\$\$ POP OMIT

BEGIN

% Structure Generating Declarations

DEFINE

OVHD = 12 % At front of each code segment

,XSTATBYTES = 254 % Read Unit Status return length

,EIGHTK = 8192

,ENDSGD =

,SYSCAP = 393216

,MAXELEMENTS = 1 % MAX ELEMENT PER DIMENSION

,MAXROWS = 48 % MAX NUMBER OF ROWS

;

FILE

CODE(KIND = TAPE, % UNKNOWN TYPE

***** LABELKIND = UNLABELLED, % READ ONLY

FILEUSE = IN,

OPTIONAL = TRUE,

NEWFILE = FALSE,

DEPENDENTSPECS = TRUE)

,LINE(KIND = PRINTER,

FILEUSE = OUT,

FRAME SIZE = 8,

MAXRECSIZE = 132)

,RMT(KIND = REMOTE,

FILEUSE = IO,

BLOCKSTRUCTURE = EXTERNAL,

FRAME SIZE = 8,

MAXRECSIZE = 132)

;

2 DIM. DIRECT EBCDIC ARRAY
3 BUFFER IMLBUF2 [0:0,0:0]
IMLBUF3 [0:0,0:0]
IMLBUF [0:0]

% To be resized
% To be resized

00141000
00142000010040006
0014220001.005.001
00142500010040006
00143000

2 DIMENSIONAL
BUFFER


```

% Get attributes of Target
RSLT := USERMAINTREQUEST(CTLUNIT,INQUIRY,23,192,0,
                          SHORTBUF,MRD,HDPRESULT);
IF RSLT > 0 THEN % Problem
BEGIN
  SHOW("MCP interface error " C
        RSLT FOR * DIGITS C
        " to get target" C
        " attributes for SCSI drive " C
        CTLUNIT FOR * DIGITS,TRUE);
  SHOWRSLT(RSLT,ATTRIBUTESV);
  SHOW("Check TARGET for problem",TRUE);
  IF NOT RELEASETARGET (OPTODO) THEN
    GO GRANDXIT
  ELSE
    GO NEXTDRIVE;
END;

SHOWINQUIRYDATA;

IF EXPRESSMODE THEN % Save for display at end of download
  REPLACE OLDFWLEVEL [0] BY TSERVOREVLVL FOR NEWFWLEVELNG;

% Get code file & match header rec. info against Target info.

IF TSERVOREVLVL = FNEWFWLEVEL FOR NEWFWLEVELNG THEN
BEGIN
  SHOW ("Servo FW levels of Target and File are the same.",TRUE);
  START "Do you still want to download the firmware? Enter YES or NO";
  PROMPT;
  IF PL NEQ "Y" THEN
    BEGIN
      SHOW ("Download will not take place for target " C
            CTLUNIT FOR * DIGITS, TRUE);
      IF NOT RELEASETARGET (OPTODO) THEN
        GO GRANDXIT
      ELSE
        GO NEXTDRIVE;
    END
  ELSE REPLACE SAVEFWLVL [0] BY 48"00" FOR OLDFWLEVELNG; % NO FORMAT
  END
ELSE
IF NOT COMPSERVOFWLVL THEN % COMPARE FW LEVELS OF FILE VS. TARGET
  IF NOT RELEASETARGET (OPTODO) THEN
    GO GRANDXIT
  ELSE
    GO NEXTDRIVE;

% If drive needs to be formatted after firmware download, get
% permission before downloading the code.
% Each FW level area in file = 8 bytes. If byte 0 neq 48"00", drive
% must be formatted after code is loaded.
IF SAVEFWLVL [0] NEQ 48"00" THEN % Drive needs formatting
  IF NOT OKTOFORMAT THEN
    IF NOT RELEASETARGET (OPTODO) THEN
      GO GRANDXIT
    ELSE
      GO NEXTDRIVE;

% Request function

IF NOT EXPRESSMODE THEN
  SHOW("Starting to download code to drive " C
        CTLUNIT FOR * DIGITS,TRUE);
IF FCODEBYTES > SYSCAP THEN
BEGIN
  NUMBROFIOS := FCODEBYTES DIV EIGHTK;
% NUMBROFIOS := * + 1;
% SHOW(" NUMBROFIOS = " C NUMBROFIOS FOR * DIGITS,TRUE);

```

```

0150201001.005.001
0150202001.005.001
0150203001.005.000
0150204001.005.001
0150205001.005.001
0150206001.005.001
0150207001.005.001
0150208001.005.001
0150209001.005.001
0150210001.005.001
0150211001.005.001
0150212001.005.001
0150213001.005.001
0150214001.005.001
0150215001.005.001
0150216001.005.001
0150217001.005.001
0150218001.005.001
0150219001.005.001
0150220001.005.000
0150221001.005.001
0150222001.005.001
0150223001.005.000
0150224001.005.001
0150225001.005.001
0150226001.005.001
0150241001.005.001
0150242001.005.000
0150243001.005.001
0150244001.005.000
0150245001.005.001
0150246001.005.001
0150247001.005.001
0150248001.005.001
0150249001.005.001
0150250001.005.001
0150251001.005.001
0150252001.005.001
0150253001.005.001
0150254001.005.001
0150255001.005.001
0150256001.005.001
0150257001.005.001
0150258001.005.001
0150259001.005.000
0150260001.005.001
0150261001.005.001
0150262001.005.001
0150263001.005.001
0150264001.005.001
0150265001.005.001
0150266001.005.001
0150267001.005.001
0150268001.005.001
0150269001.005.001
0150270001.005.001
0150271001.005.001
0150272001.005.001
0150273001.005.001
0150274001.005.001
0150275001.005.001
01503000
01504000
01504500010020003
01505000010020003
01506000010020003
0150610001.005.000
01506200010040006
01506300010040006
0150632001.005.000
0150635001.005.000

```

DECISION
 ON
 SYSTEM
 BYTES

```

SIZEOFLSTIO := FCODEBYTES MOD EIGHTK;
SHOW(" SIZEOFLSTIO = " C SIZEOFLSTIO FOR * DIGITS,TRUE);
FIRSTTIME := TRUE;
OFFSET := 0;
I := 0;
J := 0;
DO
  BEGIN
    IF J <= 47 THEN
      BEGIN
        RSLT := USERMAINTREQUEST(CTLUNIT,DOWNLOADMODE7,8192,OFFSET,0,
          IMLBUF2[I,*],MRD,HDPRESULT);
        SHOW("1I = " C I FOR * DIGITS,TRUE);
        SHOW("1OFFSET = " C OFFSET FOR * DIGITS,TRUE);
        IF RSLT > 0 THEN
          BEGIN
            SHOWRSLT(RSLT,LOADSLAVEIMLV);
            SHOWMRDBITS;
            SHOWHDPRESULT;

            SHOW("<< Microcode(1) NOT loaded!! >>",TRUE);
            IF NOT RELEASETARGET (OPTODO) THEN
              GO GRANDXIT;
          END;
        END
      ELSE
        BEGIN
          IF FIRSTTIME THEN
            I := 0;
            FIRSTTIME := FALSE;
            RSLT := USERMAINTREQUEST(CTLUNIT,DOWNLOADMODE7,8192,OFFSET,0,
              IMLBUF3[I,*],MRD,HDPRESULT);
            SHOW("2I = " C I FOR * DIGITS,TRUE);
            SHOW("2OFFSET = " C OFFSET FOR * DIGITS,TRUE);
            SHOW("2J = " C J FOR * DIGITS,TRUE);
            IF RSLT > 0 THEN
              BEGIN
                SHOWRSLT(RSLT,LOADSLAVEIMLV);
                SHOWMRDBITS;
                SHOWHDPRESULT;

                SHOW("<< Microcode(2) NOT loaded!! >>",TRUE);
                IF NOT RELEASETARGET (OPTODO) THEN
                  GO GRANDXIT;
                END;
              END;
            NUMBROFIOS := * - 1;
            OFFSET := * + EIGHTK;
            I := * + 1;
            J := * + 1;
          END
        UNTIL NUMBROFIOS = 0;
        SHOW(" NUMBROFIOS = " C NUMBROFIOS FOR * DIGITS,TRUE);
        SHOW(" I = " C I FOR * DIGITS,TRUE);
        SHOW(" J = " C J FOR * DIGITS,TRUE);
        SHOW(" OFFSET = " C OFFSET FOR * DIGITS,TRUE);
        IF J < 47 THEN
          BEGIN
            RSLT := USERMAINTREQUEST(CTLUNIT,DOWNLOADMODE7,8192,OFFSET,0,
              IMLBUF2[I,*],MRD,HDPRESULT);
            SHOW("3I = " C I FOR * DIGITS,TRUE);
            SHOW("3OFFSET = " C OFFSET FOR * DIGITS,TRUE);
            IF RSLT > 0 THEN
              BEGIN
                SHOWRSLT(RSLT,LOADSLAVEIMLV);
                SHOWMRDBITS;
                SHOWHDPRESULT;

                SHOW("<< Microcode(3) NOT loaded!! >>",TRUE);
                IF NOT RELEASETARGET (OPTODO) THEN
                  GO GRANDXIT;
              END;
            END
          END
        END
      END
    END
  END

```

```

01506400010040006
0150640201.005.000
0150640401.005.000
01506410010040006
01506415010040006
0150641601.005.000
01506420010040006
01506430010040006
0150643501.005.000
0150643601.005.000
01506440010040006
0150645001.005.000
0150645201.005.000
0150645401.005.000
0150645601.005.000
0150645801.005.000
0150646001.005.000
0150646201.005.000
0150646401.005.000
0150646601.005.000
0150646801.005.000
0150647001.005.000
0150647201.005.000
0150647401.005.000
0150647601.005.000
0150647801.005.000
0150648001.005.000
0150648201.005.000
0150648401.005.000
0150648601.005.000
0150648801.005.000
0150649001.005.000
0150649201.005.000
0150649401.005.000
0150649601.005.000
0150649801.002.006
0150650001.002.006
0150650201.002.006
0150650401.002.006
0150650601.002.006
0150650801.002.006
0150651001.005.000
0150651201.002.006
0150651401.002.006
0150651601.002.006
0150651801.005.000
0150652001.002.006
0150652201.002.006
0150652401.002.006
0150652601.005.000
0150652801.002.006
0150653001.005.000
0150653201.005.000
0150653401.005.000
0150653601.005.000
0150653801.005.000
0150654001.005.000
0150654201.005.000
0150654401.005.000
0150655001.005.000
0150655201.005.000
0150655401.005.000
0150655601.005.000
0150655801.005.000
0150656001.005.000
0150656201.005.000
0150656401.005.000
0150656601.005.000
0150656801.005.000
0150657001.005.000
0150657201.005.000

```

Choose 1st Two-Dimens
Array
393 216 Bytes

Choose 2nd Two
Dimens. Array
393 216 Bytes

```

END;
END
ELSE
BEGIN
  RSLT := USERMAINTREQUEST(CTLUNIT, DOWNLOADMODE7, 4224, OFFSET, 0,
    IMLBUF3[I,*], MRD, HDPRESULT);
  SHOW("4I = " C I FOR * DIGITS, TRUE);
  SHOW("4OFFSET = " C OFFSET FOR * DIGITS, TRUE);
  SHOW("4J = " C J FOR * DIGITS, TRUE);
  IF RSLT > 0 THEN
    BEGIN
      SHOWRSLT(RSLT, LOADSLAVEIMLV);
      SHOWMRDBITS;
      SHOWHDPRESULT;

      SHOW("<< Microcode(4) NOT loaded!! >>", TRUE);
      IF NOT RELEASESETARGET (OPTODO) THEN
        GO GRANDXIT;
    END;
  END;
END
ELSE
RSLT := USERMAINTREQUEST(CTLUNIT, LOADSLAVEIMLV, FCODEBYTES, 0, 0,
  IMLBUF, MRD, HDPRESULT);
  SHOW("5I = " C I FOR * DIGITS, TRUE);
  SHOW("5OFFSET = " C OFFSET FOR * DIGITS, TRUE);
  IF RSLT > 0 THEN
    BEGIN
      SHOWRSLT(RSLT, LOADSLAVEIMLV);
      SHOWMRDBITS;
      SHOWHDPRESULT;

      SHOW("<< Microcode(5) NOT loaded!! >>", TRUE);
      IF NOT RELEASESETARGET (OPTODO) THEN
        GO GRANDXIT;
      ELSE
        GO NEXTDRIVE;
    END;
  END;

  IF EXPRESSMODE THEN
    SHOW("Download complete. Waiting 20 seconds for prom burn to " C
      CTLUNIT FOR * DIGITS, TRUE)
  ELSE
    SHOW("Download complete. Waiting 60 seconds for prom burn to " C
      CTLUNIT FOR * DIGITS, TRUE);
    SHOW("Do not power off or alter drive " C
      CTLUNIT FOR * DIGITS, TRUE);

  % The SCSI disk drive has now turned off its SCSI interface.
  % It won't come alive until after it has done the power up
  % confidence tests (approx. 60 seconds). The second ATTRIBUTES command
  % can then be issued to display the new firmware level. NOTE: If the
  % second ATTRIBUTES command is issued before the target sequences
  % (powers itself back up) the target will hang and the program will show
  % an error.

  IF NOT EXPRESSMODE THEN
    SHOW("00:10 - Waiting for prom burn to target " C
      CTLUNIT FOR * DIGITS, TRUE);
    TIMER := 0;

  IF NOT EXPRESSMODE THEN
    DO BEGIN
      WHEN(10);
      N := (TIMER:=*+1)*10 + 10;          % Seconds
      IF NOT EXPRESSMODE THEN
        SHOWRSLT(RSLT, -N);
      END UNTIL N >= 1*60;                % Drop dead time

      IF EXPRESSMODE THEN
        WHEN (20);
        % WAIT 20 SECONDS ONLY
    END
  END

```

```

0150657401.005.000
0150657601.005.000
0150657801.005.000
0150658001.005.000
0150658201.005.000
0150658401.005.000
0150658601.005.000
0150658801.005.000
0150659001.005.000
0150660001.005.000
0150662001.005.000
0150664001.005.000
0150666001.005.000
0150668001.005.000
0150670001.005.000
0150672001.005.000
0150674001.005.000
0150676001.005.000
0150678001.005.000
0150680001.005.000
0150682001.002.006
0150684001.002.006
01507000
01508000010040006
0150820001.005.000
0150840001.005.000
01509000
01510000
01511000
01512000
01513000
01514000
0151500001.005.000
01516000
01517000
01518000
01519000
01520000
01521000
01521200010020003
01521400010020003
01521600010020003
01521800010020003
01522000010020003
01523000010020003
01524000
01525000
01526000
01527000
01528000
01529000
01530000
01531000
01532000
01533000
01534000
01534500010020003
01535000010020003
01536000010020003
01537000
01538000
01538500010020003
01539000
01540000
01541000
01541500010020003
01542000010020003
01543000
01543200010020003
01543400010020003
01543600010020003

```

LOAD SINGLE
ARRAY